**Submitted By:**

**Alharbi Theyab**

**Hastimal Jangid**

**Neha Choudhary**

**Wuchen Wang**

**First Increment Report**

I. Importing Existing Services/API

At current point of the time, we are not using any external/existing services or API. However, we were planning to include Endnote API in the future increments to allow researchers to cite research papers they found relevant to their submitting documents (Endnote). Potentially, telegram open source API for chatting service could also be included in future increments to allow instant messaging among researchers (telegram).

II. Detailed Design of Services

As detailed in our project plan, the design of services in this project was primarily focused on data retrieval and data storage and privacy. The web application should be capable of providing secure data storage and retrieval for the researchers and collaborators to use. Therefore, our design of these at this stage of development partially relies on the design of transfer classes, namely, beans in our applications. We have also designed the database access objects for each of those transfer classes in the application server.

1) User stories/cases.

In the first increment, we have created and followed these story guidelines to finish our respective work from ScrumDo:

Story #2: In order to create the data storage, data retrieval service, we need to work on the frontend webpages so that we can have the forms for collecting user input.

Story #3: In order to create the data storage, data retrieval service, and privacy service, we need to build the backend components so that they will handle the persistence of data in MongoDB.

Story #4: In order to improve the usability of the frontend, we need to write test cases to make sure frontend responds to backend objects so that the communication between them are integral.

Story #5: In order to deal with the hardest part for the team first, we need to tackle the MongoDB, create schemas in database so that everything else to be done will be less difficult.

Story #6: In order to generate a prototype system, we need to work on frontend and backend simultaneously and create mock tests to validate the input and output so that we can guarantee the components we write are working early on.

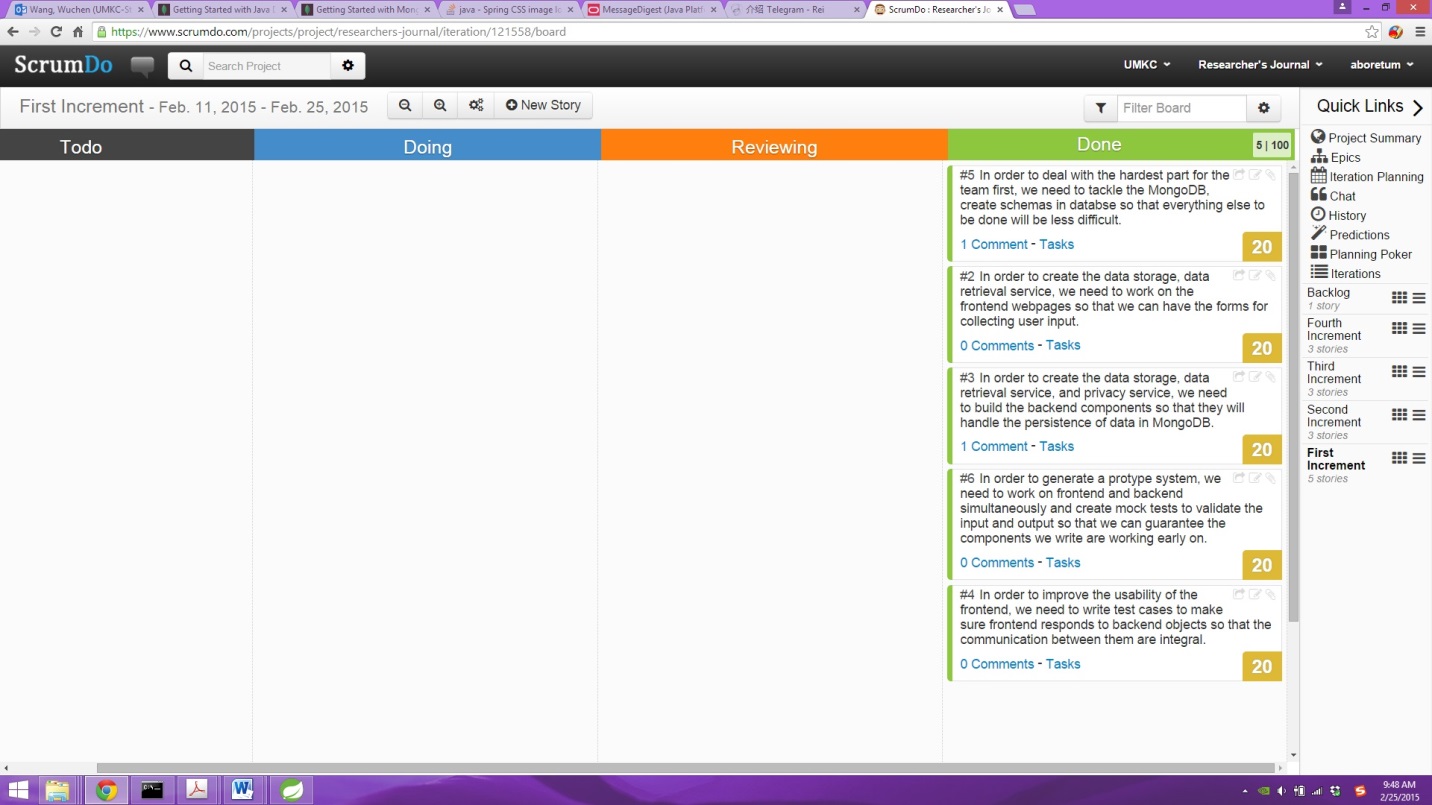


Figure 1. ScrumDo dashboard for first increment.

2) Service description

The service designed in this increment is mainly the user registration and user login service, which falls into the category of general data storage and retrieval service. In addition, the registration put the user password in database in a secured manner, and the login service also authenticates the user password, which contribute to the privacy service discussed in project plan. The major task of privacy service regarding group content separation will be implemented in second increment and the module of authentication designed in first increment could be reused.

3) Service sequence diagram

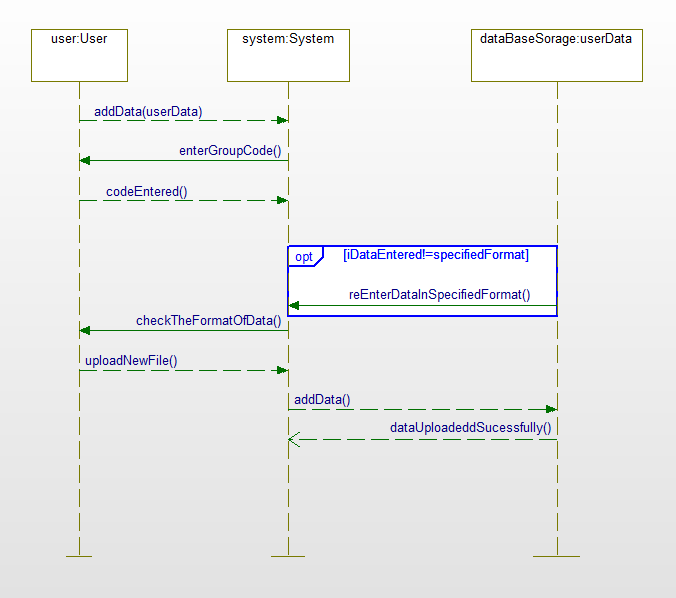


Figure 2. Data storage sequence diagram (from project plan).

4) Service class diagram

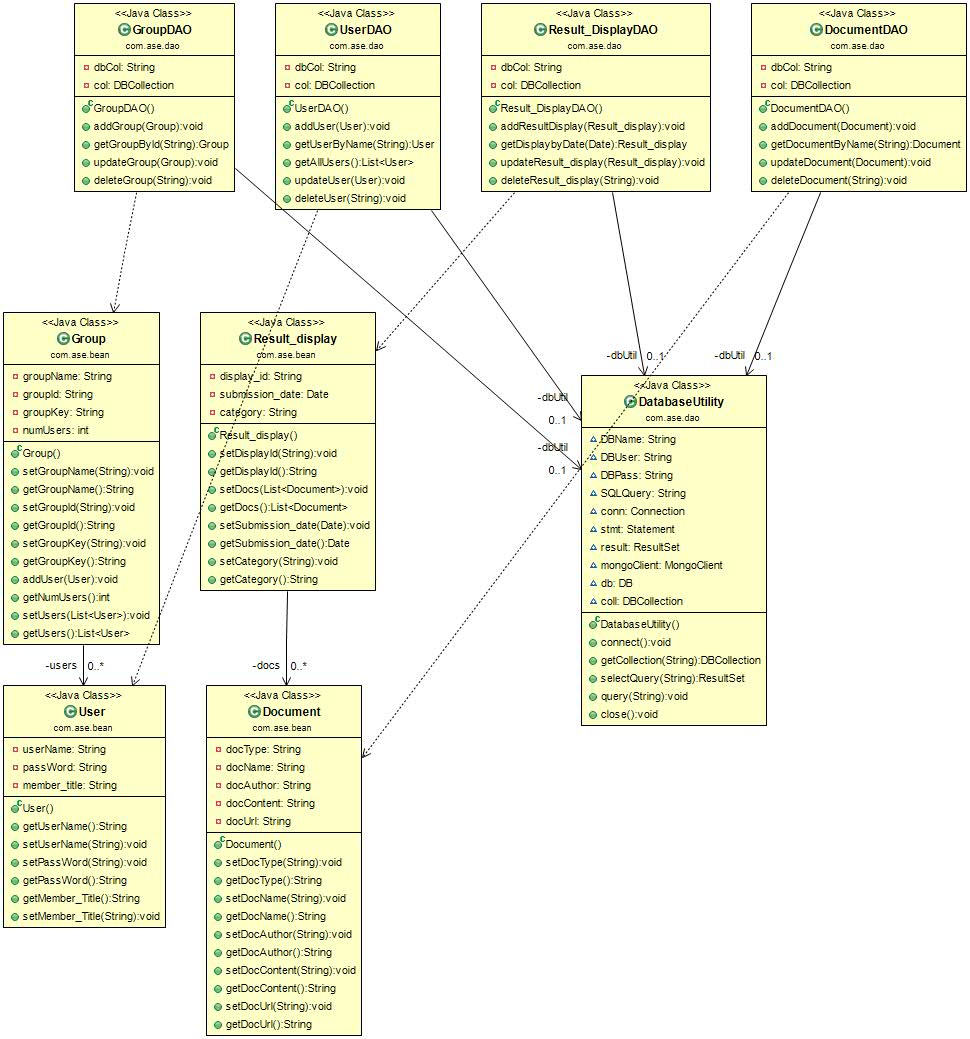


Figure 3. UML class diagram for server-side java beans and DAO design using ObjectAid UML explorer.

5) Design of mobile interface

Not applicable to this project.

6) Design of Unit test cases

Three test cases for AddUserController.java, UserDAO.java and LoginController.java were created and the methods of those classes were fully tested. The test cases were created using JUnit, as it is suitable for writing tests for Java projects.

III. Implementation

1) Implementation of rest services.

The services of user registration and user login are strictly implemented as REST services. The registration point to the LoginController class in the application server, which is implemented by the request mapping of url “register” to the resources available for user registration. The login service point to the AddUserController class, which is implemented by request mapping of url “login”, with the HTTP post method handling user input of user name and password.

2) Implementation of user interfaces.

The user interface was implemented by frontend webpages, which was written in HTML5+CSS+JavaScript.

3) Implementation of test cases.

The three test cases for relevant classes were created with JUnit, and the corresponding methods were tested by JUnit Assert class static methods.

IV. Testing

Extra testing was performed using basic system print line methods to verify the authentication process and other methods not included in the two main classes responsible for user registration and login.

V. Deployment

1) Project ScrumDo link:

<https://www.scrumdo.com/projects/project/researchers-journal/iteration/121558/board>

2) GitHub repository URL:

<https://github.com/aboretum/Researcher-s-Journal>

VI. Report

1. We have designed the frontend for user login and registration, as shown in Figure 4,5.

2. The backend connections to server through transfer classes and data access objects, as shown in Figure 3. We used ObjectAid UML explorer to generate the backend class diagram from the source code, as the software support this reverse engineering functionality. So the class diagram in Figure 3 represents both the design and outcome of backend classes.

3. As shown in Figure 6, the mongoDB database for the project was installed and created. The Java connectivity driver for mongoDB was installed and tested.

4. The registration and login service in the backend was fully integrated with frontend, as shown in Figure 7.

5. The MD5 hash function were utilized to create message digest of user password to be stored in database, as shown in Figure 8, the user password was stored as a new message digest text.

6. The relevant classes for login and registration service were tested with JUnit, as shown in Figure 9.

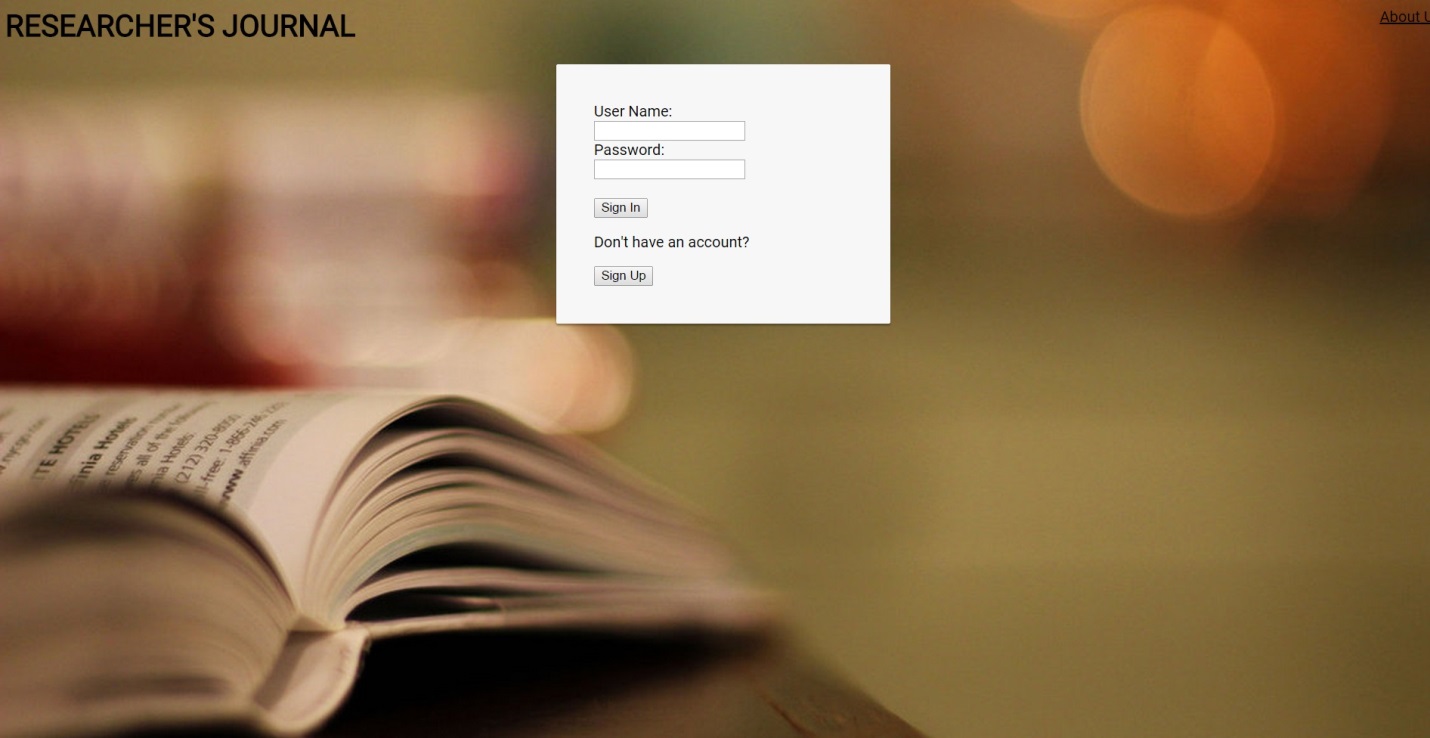
Figure 4. Login page created for the web app.



Figure 5. Signup page created for the web app.

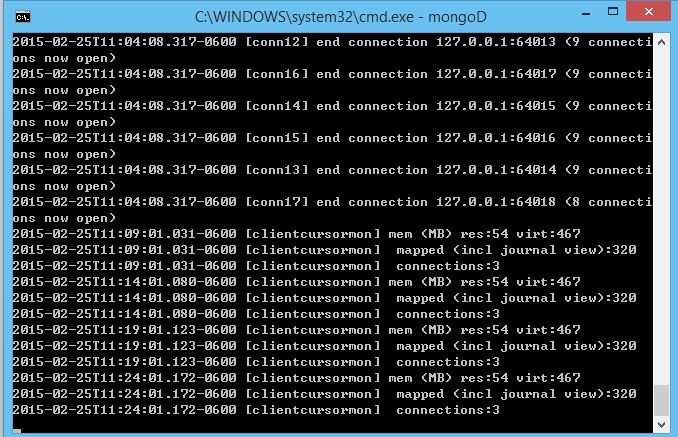


Figure 6. A running instance of mongoDB database.

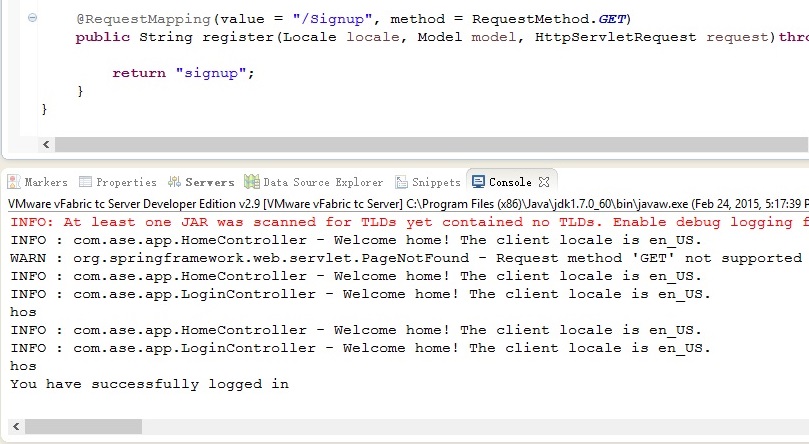


Figure 7. Integration of login service with frontengpage.

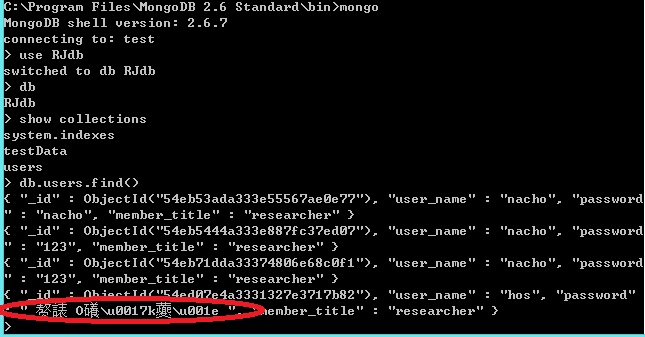
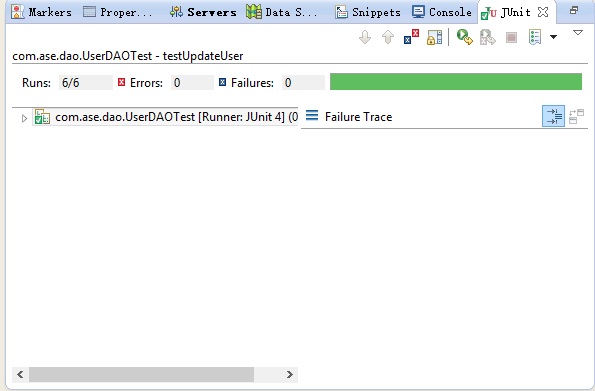


Figure 8. Storage of user data in mongoDB collection, with user password stored as message digest.



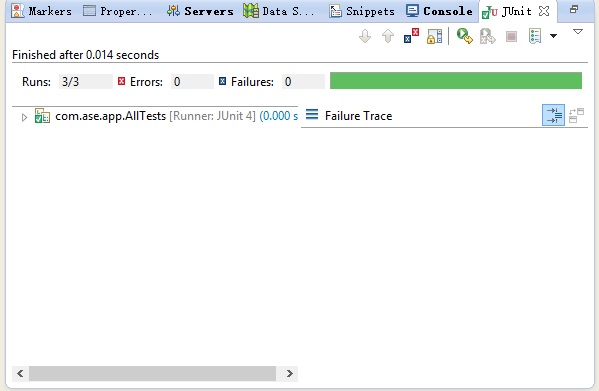


Figure 9. Test cases and test suites passed in JUnit test.

VII. Project Management

- Implementation status report

Work completed:

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Task** | **Assignee** | **Estimated time taken** |
| 1 | Created frontend pages, including login, registration, and main page. | Neha,  Theyab | 8 hours |
| 2 | Created backend components, transfer classes, data access objects. Created database connectivity to MongoDB. | Wuchen,  Hastimal | 5 hours |
| 3 | Created secure login service using MD5 hash function to store message digest of user provided password in database. | Wuchen,  Hastimal | 5 hours |
| 4 | Testing of frontend components | Neha,  Theyab | 2 hours |
| 5 | Testing of backend services. | Wuchen,  Hastimal | 2 hours |

Work to be completed:

The planned work in ScrumDo for second increment includes design and implementation of privacy services, automatic layout of results documents. This work will be distributed to frontend and backend again to respective team members.

Issues and concerns:

We planned to finish the design of the main page in the first increment, however, the time is limited so we have not yet finished a working design. We will solve this in the next increment by employing professional bootstrap themes and adapt to our needs, this could save us a lot of time. We have fully realized that to build everything of frontend from scratch is a big challenge.

We are not using any external APIs for our project at this stage, we will include those in our next increments. One concern about the endnote API is the requirement for license, for which we are actively asking for uncommercial usage from the vendor at this time.

VIII. References

1. <http://endnote.com/product-details/integration>

2. <https://core.telegram.org/api>